# SPIRAL MODEL PILOT PROJECT

# INFORMATION MODEL

WP-2210/0025/0001/00
October 28, 1991

Contract # NAS8-37680

Prepared For:
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Information and Electronic Systems Laboratory
Systems Software Branch
Marshall Space Flight Center, Alabama 35812

Prepared By:
Nichols Research Corporation
4040 South Memorial Parkway
Huntsville, Alabama 35815

## TABLE OF CONTENTS

## 1.0 INTRODUCTION

In July 1991, a three month development task was initiated to accomplish two objectives:

1) The development of an Environmental Control and Life Support System (ECLSS) Process Control Prototype system for evaluation of technologies related to that program including use of Graphical User Interfaces (GUI), Application Generators, networked control and monitor stations, and control algorithms.

2) An evaluation of the Spiral Model development approach to allow Marshall Space Flight Center (MSFC) to develop an experience base of that software management methodology.

This paper presents a discussion of the Information Model that was used as part of the Spiral Model methodology.

A key concept of the Spiral Model is the establishment of an Information Model to be used by management to track the progress of a project. The Information Model is the set of metrics that is to be measured and reported throughout the life of the project. These metrics measure both the product and the process to ensure the quality of the final delivery item and to ensure the project met programmatic guidelines (i.e. cost, schedule, deliverables, computer utilization, etc.). The beauty of the Spiral Model, along with the Information Model, is the ability to measure not only the correctness of the specification and implementation of the requirements but to also obtain a measure of customer satisfaction.

Because of the limited resources of this project, certain information was not collected. However, a goal of the project was to fully define the information that could and should be collected for projects of this nature.

The following areas will be discussed in this paper:

1) Project overview.

2) Risks and demonstrations.

3) Software metrics.

4) Project documentation.

5) Project conclusions.

## 2.0 APPLICABLE DOCUMENTS

### 2.1 Reference Documents

The following documents of the exact issue shown form a part of this Information Model to the extent specified herein.

| | |
|---|---|
| MM 8075.1 | MSFC Software Management and Development Requirements Manual, Software and Data Management Division. |
| SEL-84-101 | Manager's Handbook for Software Development, Revision 1, Software Engineering Laboratory Series, November 1990, NASA/GSFC. |
| WP-2210/0025/0001/00 | ECLSS Process Control Prototype Software Management Plan, July 29, 1991. |
| WP-2210/0024/0001/01 | ECLSS Process Control Prototype Requirements and Design Specification, Spiral Model Pilot Project, September 26, 1991. |

## 3.0 PROJECT OVERVIEW

### 3.1 Spiral Model Pilot Project Overview

The purpose of the Spiral Model Pilot project was to evaluate the use of the Spiral Model life-cycle as applied to the development of the ECLSS Process Control Prototype. The ECLSS Process Control Prototype was developed using guidelines defined in MSFC Software Management and Development Requirements (SMADR) Manual, MM 8075.1, and Barry Boehm's risk-driven approach to the software development process.

Per MM 8075.1, the ECLSS Process Control Prototype is classified as a Category C project. Category C projects are small, simple, and have a low criticality. They are usually conducted within a self-contained organization, do not involve complicated interactions with other projects or life-cycles, and are not on the critical path for any other development effort.

The ECLSS Process Control Prototype was a small project with significant risks in technical feasibility and user acceptance and with requirements which changed throughout the life of the project. For projects with this profile, the SMADR recommends either a phased delivery or an incremental delivery process model, with prototyping (i.e. the use of the Spiral Model approach).

The ECLSS Process Control Prototype project was chosen as a testbed for the Spiral Model for several reasons: 1) the requirements for the project were loosely defined at the start of the project and were expected to change as the project matured, 2) the hardware and software available for the project were limited and were expected to change, and 3) there was a limited staff for development and test of the product. A primary intent of the project was the utilization of existing software products (e.g. development tools, code generators, networking software, etc.) that were to be integrated to achieve the desired requirement.

### 3.2 ECLSS Process Control Prototype Overview

The ECLSS Process Control Prototype consisted of three distinct systems:

1) Water Recovery Control System

2) Water Recovery Monitoring System

3) Data Transfer Process

The initial and unchanged requirement of the ECLSS Process Control Prototype was to demonstrate a control system for the Water Recovery system. Control was accomplished using a computer based manual control function that interfaces to controller software to

perform predefined control functions. A Water Recovery Monitoring System provided a monitor of the hardware activity data, sensor data, and current system control state data. The physical monitor station was separate from the control station. The hardware activity data was transmitted over a network to the real-time monitoring station. Figure 3.2-1 depicts the ECLSS Process Control Prototype.

### 3.3 Project Development Process Overview

The ECLSS Process Control Prototype was developed using the following development and Computer-Aided Software Engineering (CASE) tools:

1) Application Generator (AG) - The AG is a control engineering development application and code generator developed by Integrated Systems, Inc. The AG provides an environment for the development of control algorithms, simulation and testing of control algorithms, and automated generation of controller Ada software.

2) Transportable Application Environment Plus (TAE+) - TAE+ is a GUI development tool produced by NASA, Goddard Space Flight Center (GSFC). TAE+ provides a toolset for graphically developing interactive displays and generating the executable software.

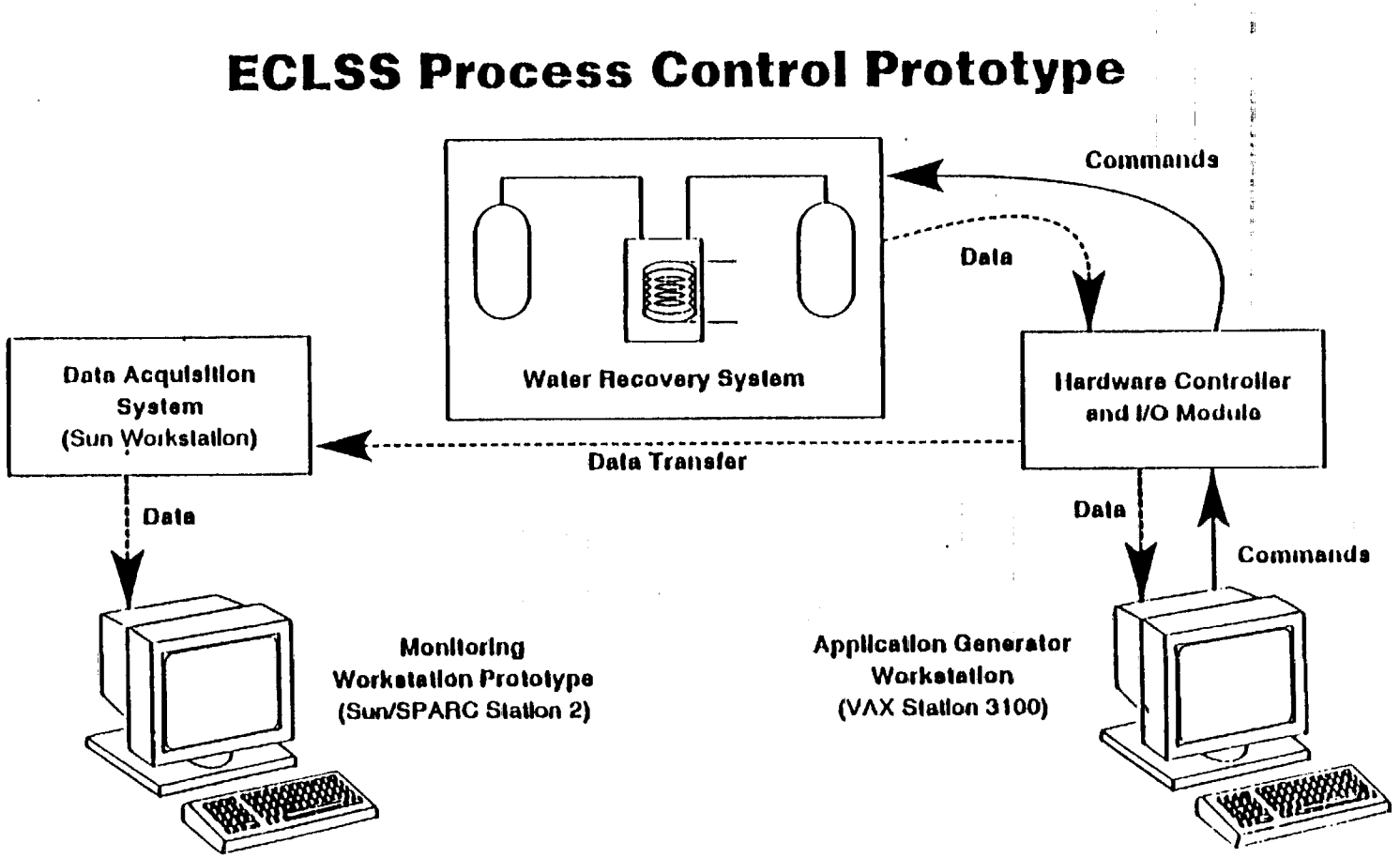
ECLSS Process Control Prototype
Commands
Data
Water Recovery System
Data Acquisition System (Sun Workstation)
Hardware Controller and I/O Module
Data Transfer
Data
Data
Commands
Monitoring Workstation Prototype (Sun/SPARC Station 2)
Application Generator Workstation (VAX Station 3100)

FIGURE 3.2-1

## 4.0 RISKS AND DEMONSTRATIONS

### 4.1 Project Risks

The ECLSS Process Control Prototype project was managed using the Spiral Model process. The Spiral Model methodology utilized a "risk driven" and "demonstration driven" process to guide the project to a successful conclusion.

Project success was tracked and measured by demonstrating that risks had been eliminated and system capabilities implemented. At the start of the Spiral Model life-cycle, project risks were defined and prioritized. This planning phase was an iterative process involving all team members including the development and test personnel, the customer (MSFC personnel), and the Quality Inspection personnel.

Risks were identified in areas where: 1) the project would be considered out of compliance with system requirements if a capability was not successfully implemented, 2) new technology was being utilized or developers had limited experience, 3) functionality to be developed was perceived as difficult, and 4) hardware or development tools may be unavailable. Once risks were defined, they were prioritized based on their importance to the success of the project. The Spiral Model Pilot project risks are listed in their order of priority:

1) Networking/Integration - Three different workstations were required to communicate with each other. These machines needed to be networked together and data transmitted over the network in order for the ECLSS Process Control Prototype to be fully functional. These machines had never been networked together and the developers had limited networking experience.

2) GUI Acceptability - One of the primary purposes of this project was to develop a modifiable GUI which was acceptable to the customer. The GUI was developed using TAE+. TAE+ was to be installed on a Sun/SPARC workstation for the first time and its compatibility with other interfacing workstation applications was a concern.

3) Hardware Interface - The water recovery system hardware developed for this project had never been completely interfaced with the AG controller and there were known problems with the hardware. Data was to be transferred between the controller and the TAE+ GUI for the first time using a project developed Data Transfer application. The successful implementation of this data transfer loop was essential to the project.

4) Development of Control Algorithms - The development of the water recovery system's control algorithms was also essential to the success of the project. The developer would be using the AG for the first time and would require a learning period. There was a concern that the AG's documentation, if used alone, would not provide adequate training.

   The development schedule for the control algorithms was a major concern. The developer was scheduled to leave the project in early August and therefore needed to be complete with the control algorithms early in the project. This created a very tight development schedule.

5) Equipment availability - The customer's facility, which contains all of the ECLSS Process Control Prototype hardware, was scheduled to move in August. This was expected to impact the development by at least one week. The development schedule needed to take this risk into consideration and allow for equipment unavailability or schedule slides. Again, with a tight development schedule this risk needed to be monitored throughout the project in case the move date was rescheduled.

By defining and prioritizing risks, project activities and development could be scheduled in order to reduce and eliminate risks. Major risks were divided into sub-risks, where appropriate, to develop a schedule with more manageable and smaller activities to be performed and monitored. The project was planned to have five incremental cycles. Each cycle addressed specific risks and development activities and ended with a demonstration. The demonstration became the primary measure of project success and progress. Table 4.1-1 is the project schedule displaying the risks, development activities, and demonstrations for each cycle.

## 4.2 Demonstrations

The purpose of a demonstration was to display the successful elimination of project risks and the successful implementation of system capabilities assigned to that cycle. The following checklist was reviewed and responded to during the project planning phase and prior to the start of each cycle, software build, and demonstration:

1) What would be the content, schedule, and purpose of each software build?

2) What were the risks, priority of risks, and approach for resolution?

3) What were the planned capabilities to be demonstrated at

demos?

4) How will the project progress be measured?

5) How will the design quality be measured?

It was determined by the team that a Pass/Fail Criteria would be developed and followed for each demonstration. The Pass/Fail Criteria was developed in response to the checklist discussed above. The criteria provided: 1) the purpose and date for a demonstration, 2) risks identified to be addressed during a demonstration and the planned percentage of risk reduction, 3) specific capabilities to be demonstrated by developers (operator's actions during the demonstration), 4) the system's appropriate and correct response to an operator's action displaying that a capability was functioning correctly, and 5) the pass/fail status of a specific capability which measured the quality of that capability.

The Pass/Fail Criteria basically defined a scenario to be implemented during a demonstration. The scenario displayed the elimination of project risks and the implementation of system capabilities. At later demonstrations, system requirements, from the Requirements and Design Specification, were included in the Pass/Fail Criteria and tested. Appendix 1 provides a report for each demonstration. Prior to a demonstration, a report contained the demonstration's purpose, risks to be addressed and planned percentage of completion, and the Pass/Fail Criteria. After a demonstration was held a results section was added to the report.

At most of the demonstrations, outside personnel were invited to participate in the evaluation of the project. Both management and technical personnel provided valuable comments, suggestions, criticisms, and recommendations to the project. The development team was sensitive to the reaction of those people who were not directly involved with the project. Their views were a good measure on the clarity of the requirements and the understanding of the implementations.

After each demonstration, a project review meeting was scheduled to discuss the results of the demonstration and the Pass/Fail Criteria. Failed criteria were discussed and an approach determined to resolve or pass the criteria. It was at this time that the need for a delta-demo was determined. The project requirements were reviewed at this meeting to determine necessary changes and updates. The project risks were reviewed to determine their reduction or need for re-prioritizing. If a new risk was discovered during development, it's resolution would be addressed and scheduled. The customer was a primary participant in all demonstrations, review meetings, and design and scheduling decisions to ensure that the product being developed would meet the needs and satisfaction of the customer.

Table 4.2-1 displays the planned percentage of risk reduction or elimination for each cycle. The actual percentage of reduction or elimination is also displayed for tracking project progress. The risk reduction was evaluated based on a cycle's demonstration. Actual percentage of risk reduction or elimination was not quantified for a cycle until after a delta-demo, if one was held. Delta-demos were held for cycles one and two.

From this Table, it is apparent that project development managed to stay on track although delta-demos were scheduled for cycles one and two where some slippage in planned development occurred. Delta-demos were expected in the earlier cycles due to developers requiring a learning period for tools and the fact that the project was addressing the most difficult risks first. More errors were also expected early in the project creating more failed criteria. As the project requirements matured and were more clearly defined for each cycle and developers' expertise in the use of tools increased, demonstrations were more successful with scheduled activities being completed on time and as expected.

**TABLE 4.1-1 - PROJECT SCHEDULE**

**Cycle 1 Risks, Development Activities, and Demo**

1) Networking/Integration
   a) connect the serial communications cable from the AG controller to the Sun 3/260.
   b) communicate between the AG controller and Sun 3/260 using the Data Transfer application to reside on the Sun 3/260.

2) TAE+ Installation Complete
   a) fonts need to be operational.
   b) test all TAE+ attributes.

3) Software Applications Compatibility on SPARC Workstation
   a) exercise TAE+ using Open Windows as the window manager in order to evaluate TAE+'s performance with Open Windows.

4) TAE+ C Code Modifications
   a) generate TAE+ C code.
   b) modify C code with the appropriate TAE+ calls for display manipulation.
   c) develop local test data file(s) (data will not be AG generated).
   d) connect local test data file to C code.
   e) display local test data in test display.

5) Graphical User Interface Acceptability
   a) develop and evaluate a TAE+ user interface displaying all AG transmitted values and current controller software state.
   b) gather customer opinions to begin display design.

**DEMO 1 (May 31, 1991)**
- AG network connectivity.
- User Interface and local test data file on the SPARC workstation.

**Cycle 2 Risks, Development Activities, and Demo**

1) AG Controller and Hardware Interface
   a) interface current controls to hardware for experience.
   b) augment current controller data acquisition software.

**TABLE 4.1-1 - PROJECT SCHEDULE (cont.)**

2) Data Transfer (physically move controller data across the network)
   a) define data format to be transferred, read, and displayed at the SPARC workstation.
   b) move data from AG controller to C application.
   c) move data from Sun 3/260 to SPARC displays.
   d) move data from AG controller to Sun 3/260 to SPARC workstation.

3) C Application, residing on the Sun 3/260, for Data Transfer
   a) develop C application to transfer raw AG controller or hardware data to TAE+.

**DEMO 2 (June 28, 1991)**
- Hardware test data transfer across ECLSS control loop (hardware, AG controller, Sun 3/260, SPARC).
- Exercise hardware via manual control box.

**Cycle 3 Risks, Development Activities, and Demo**

1) Develop new control algorithms for AG controller software
   a) simulation.
   b) development of control algorithms.
   c) development of user interface.
   d) generate Ada code, compile, and download.
   e) exercise hardware using Interactive Animation.

**DEMO 3 (August 2, 1991)**
- Final AG control of Water Recovery System hardware.

**Cycle 4 Risks, Development Activities, and Demo**

1) Complete SPARC Workstation User Interface (Graphical User Interface)
   a) final design of Graphical User Interface.
   b) develop.
   c) test.

**DEMO 4 (August 20, 1991)**
- Final hardware data transfer.
- Water Recovery System data being monitored on the SPARC workstation's Graphical User Interface.

## TABLE 4.1-1 - PROJECT SCHEDULE (cont.)

**Cycle 5 Risks, Development Activities, and Demo**

1) Final system updates for complete system operability.

**DEMO 5 (September 9, 1991)**
- Final system operability.

**TABLE 4.2-1 - RISK REDUCTION AND ELIMINATION**

| | CYCLES: | | | | |
|---|---|---|---|---|---|
| RISKS: | Cycle 1 | Cycle 2 | Cycle 3 | Cycle 4 | Cycle 5 |
| 1) Networking/Integration | | | | | |
| Planned | 100% | | | | |
| Actual | 100% | | | | |
| 2) Graphical User Interface | | | | | |
| Planned | 48% | 76% | 84% | 92% | 100% |
| Actual | 18% | 54% | 84% | 90% | 100% |
| 3) Hardware Interface | | | | | |
| Planned | 40% | 100% | | | |
| Actual | 40% | 90% | 100% | | |
| 4) Development of Control Algorithms | | | | | |
| Planned | | 40% | 100% | | |
| Actual | | 30% | 100% | | |

## 5.0 SOFTWARE METRICS

Development of software metrics has received a lot of attention by industry, Department of Defense (DOD), and NASA. If there are any standard metrics, certainly those specific in the Manager's Handbook for Software Development, Revision 1 (Software Engineering Laboratory Series, SEL-84-101, November 1990, NASA/GSFC) would constitute a good set. The key eight metrics identified are listed and discussed below relative to the Spiral Model task.

1) Source code growth rate.
2) Effort data.
3) System size estimates.
4) Computer usage.
5) Error rates.
6) Reported/corrected software discrepancies.
7) Rate of software change.
8) Development activity status data.

These areas of metric collection have been considered for their benefit in measuring the progress of the Spiral Model Pilot project or a similar project. To determine the usefulness of a specific type of metric, project considerations need to be made. Some considerations applicable to the Spiral Model Pilot project are:

1) The project was a prototype development, therefore, changes were encouraged and expected.

2) Development tools, specifically a code generator, were utilized for almost all of the project development.

3) The project did not require complicated interactions with other projects or organizations and was not on the critical path for any other development effort.

4) The project team was small, consisting of five people, therefore monitoring of project activities was easily managed.

While metrics should be maintained and reported, the collection can be a significant undertaking, therefore, the benefit of collecting a specific metric needs to be weighed against the cost and time of collecting the metric. It is important for each data item collected to serve a well-defined purpose. Taking this into account along with the project considerations, the following discussion examines the need for collecting metrics in the eight categories defined by SEL. It is important to note that these considerations were not made prior to the start of the Spiral Model Pilot project and metrics were not collected in these categories on the Spiral Model Pilot project unless otherwise noted. This discussion is for the benefit of "lessons learned" and "what should be done for the next project of this nature".

## 5.1 Standard Key Metrics

### 5.1.1 Source Code Growth Rate

If source code growth rate was to be monitored, then prior to project development the source lines of code (SLOC) would be estimated for a complete system. The total SLOC would then be broken into the number of developed lines of code which should exist at a point in time or milestone. The actual number of SLOC at the milestone point would then be used to determine project progress.

The Spiral Model Pilot project developed a prototype system in an iterative process where the product was reviewed at the end of a cycle and modifications determined. Because changes were encouraged and expected, SLOC were not estimated prior to development nor did the project place limitations on the size of the product at a point in time or at the end of the project.

The use of code generators during development also eliminated the need to monitor progress based on SLOC. Automated generated code may be more verbose than human generated code, thus producing more SLOC. (While automated generated code may be larger and a slight run-time performance degradation may occur, the benefits such as uniformity, no coding errors, speed, and cost effectiveness should also be noted.) Developing a prototype with automated code generators also does not allow a lot of control over developed SLOC, therefore, recording source code growth rate may not benefit a project of this type. In the future, as more data is collected on the use of automated code generators and the quality of code generated, SLOC may be an appropriate measurement.

Of interest to the reader, at each of the demonstrations the question on the efficiency of the AG was asked. While this is an interesting question, it was not an objective of this project to perform any analysis on the merits of the AG.

An important consideration in the past for measuring the source code growth rate metric has been the limited amount of memory available for the target computer system. For this project there was never any concern on availability of memory.

### 5.1.2 Effort Data

Effort data could have been collected and monitored for the Spiral Model Pilot project to determine the completeness of project requirements and adequacy of developers' expertise in the use of development tools. If a lot of effort was being spent on re-defining requirements, then the planning phase or requirements definition phase was inadequate. If developers are having difficulty accomplishing activities, then additional training for development tools may be needed, manpower may need to be increased,

or they do not have a clear understanding of requirements, again indicating the initial planning phase was inadequate.

The initial planning phase for any Spiral Model life-cycle should be heavy in man-hours. This phase requires effort by management and developers in defining project risks, activities, cycles, and demonstrations. A good understanding of the product to be developed is required at this stage in order to define requirements. Once the planning phase is complete, management hours may decrease while development hours are at a full-time status. At the end of a cycle, the product is reviewed and modifications made. A lot of changes may indicate that requirements were poorly defined or developers have a poor understanding of requirements. If meetings are being required during the cycle to review or clarify requirements, then the planning phase may have been inadequate. For the Spiral Model Pilot project, the development of a prototype encouraged some changes to requirements between cycles in order to meet the needs of the customer.

An important consideration for the Spiral Model approach was an almost continuous discussion and evaluation of the progress and quality of the project. Requirements were continually evaluated and solutions or implementations discussed thoroughly before being finalized. Impacts on project risks and total requirements were understood during each spiral.

### 5.1.3 System Size Estimates

As stated in section 5.1.1, CASE tools and automated code generators were utilized, therefore, control over the size of the system was not applicable and would not be appropriate for measuring project progress. The project team also did not want to place limitations on the size of the system and did not attempt to utilize the features of the CASE tools to measure system size. However, system size for the GUI was 2091 kilobytes, the control application was 867 VAX blocks, and the Data Transfer application was 24 kilobytes.

### 5.1.4 Computer Usage

Computer usage is generally measured in Central Processing Unit (CPU) hours and compile time. The Spiral Model Pilot project's development environment consisted of CASE tools and code generators, thus eliminating the need to measure computer usage.

### 5.1.5 Error Rates

Error rates could be measured for the Spiral Model Pilot project by tracking failed criteria at demonstrations. While error rates were not collected, a project profile could be developed based on demo reports contained in Appendix 1. A project profile would

display a decrease in errors as the project progressed and a quick resolution of errors.

It is important to note that not all failed criteria were due to errors, but due to a capability being unavailable at a demonstration. The demo report should clarify the cause for a criteria failing. A criteria may fail because: 1) the demonstrated capability does not work properly, does not work completely, or does not meet requirements or 2) the capability is unavailable for the demonstration.

The SEL explains that error rates should decrease on a project from phase to phase. The decrease, of course, should result from developers' increased knowledge of the development environment and requirements and the development of reliable software, not from inadequate testing. Errors on the Spiral Model Pilot project did decrease from demonstration to demonstration. Errors early on were caused by unstable requirements, misunderstanding in the intent of a requirement, and insufficient expertise in a development tool. As requirements became more stable and developers' knowledge of tools increased, errors decreased. Testing also became more vigorous as the project progressed. Errors were always quickly resolved either by a delta-demo or by the next development cycle.

### 5.1.6 Reported/Corrected Software Discrepancies

Reported and corrected software discrepancies could have also been monitored on the Spiral Model Pilot project based on a demonstration's Pass/Fail Criteria. During a cycle, if a failed criteria existed, it was included in that cycle's delta-demo or in the next cycle's demonstration and tracked by a Pass/Fail Criteria. Software discrepancies or failed criteria were dropped from the Pass/Fail Criteria once the criteria passed. Monitoring reported software discrepancies vs. corrected or closed discrepancies would have displayed if timely closure of discrepancies were occurring, insuring that staffing was adequate and design was reliable.

A project's goal should be to resolve discrepancies as soon as they are reported. This was the case for the Spiral Model Pilot project. Discrepancies, like errors, were resolved from demo to demo or by the next cycle and the occurrence of discrepancies decreased as the project progressed.

Discrepancies were quickly resolved on the Spiral Model Pilot project due to the utilization of code generators. The project could be confident that the generated code was sound and that the error had to be in the design. Resolution was faster knowing where to look for problems.

A discrepancy report should be developed and used for tracking problems and closures. The Pilot project only relied on the Pass/Fail Criteria for tracking progress which was adequate for a

small project. With only five team members, monitoring activities and problems was not difficult or complex. A discrepancy report, however, would provide a description of the problem, how the problem is to be resolved, and the time-frame for closure. A discrepancy report, especially on larger projects, would provide a mechanism for insuring the resolution of problems.

### 5.1.7 Rate of Software Change

The Spiral Model Pilot project attempted to monitor software changes via a Change Control Board (CCB). The SEL defines software changes to include errors, however, errors detected during demonstrations were not monitored by the CCB. Changes in software requirements or hardware design were controlled by the CCB.

Changes in software and hardware were monitored to track completeness and stability of requirements. Some software or hardware changes were expected due to the development of a prototype and in fact, were encouraged. The prototype evolved as capabilities were determined to be desirable or undesirable.

Changes were reviewed to determine if a requirement had been poorly defined or if developers were deviating from baselined requirements. While deviation from baselined requirements was acceptable during a cycle's review following the demonstration, adherence to baselined requirements was important during a cycle's development phase.

A cycle's review provided an opportunity to determine if development was meeting customer's needs and progressing smoothly. The cycle's review would determine activities and changes for the next cycle. The review would also baseline requirements. Changes during a cycle's development phase could impact a demonstration's success and schedule and, therefore, required closer monitoring by the CCB.

Rate of software changes for the Spiral Model Pilot project should have also been tracked and measured by the number of changes to the requirement specification. A number of changes to the requirement specification may have indicated that the system to be developed had not been completely defined prior to development. Tracking changes to the requirement specification may have required development to stop and the project team to replan system requirements.

### 5.1.8 Development Activity Status Data

Development activity was monitored on the Spiral Model Pilot project and tracked as displayed in Table 4.1-1. Development activity was planned based on prioritized risks. Activities were defined to reduce and eliminate risks, assigned to one of the five project cycles, and due the date of the cycle's demonstration. If

an activity was not complete or not working correctly for a demonstration, a delta-demo was held or the activity was due at the next cycle's demonstration.

The SEL measures activities based on units coded, units read, and units tested. This measurement would not be appropriate for the Spiral Model Pilot project. The use of code generators eliminated the need for unit testing.

While measuring units coded was not appropriate, a measurement applicable to the development tool being used may have been applied. For instance, the AG's control algorithms are based on a hierarchy of super blocks, usually representing a system function. A super block consists of control engineering blocks representing system activity. Code templates are generated from the engineering blocks. A metric based on developed super blocks, generated and executed code for super blocks, and testing of hardware activity may be appropriate for a similar project.

The Spiral Model Pilot project did not assign completion dates for each scheduled activity. An activities completion date was the demonstration's scheduled date. Assigned completion dates for each activity may have alerted the project to trouble areas prior to a demonstration. More effort could have been applied to a potential trouble area reducing the number of errors or failed criteria at a demonstration. Delta-demos may also have been eliminated.

## 5.2 Unique Detailed Metrics

The Spiral Model team reviewed the process and products of this project to identify those metrics that could have been measured to provide additional insight into the development of the final product. These metrics are identified below and are provided to allow for an understanding of what metrics this team considered important. Future projects could consider utilization of these metrics.

**Measurements specific to the AG:**

1) Number of Superblocks - Superblocks create a decomposition of the model, hierarchy.
2) Number of Engineering blocks - Control algorithms consist of engineering blocks, code is generated from the engineering blocks.
3) Number of engineering blocks contained in a Superblock - track for quality of decomposition.
4) Number of displays - interactive animation.
5) Display icons - interactive animation icons.
6) Hardware connections - number of signals being transmitted.
7) Total generated Ada lines of code (LOC).
8) Total manually written Ada LOC.

9) Total Ada commands.
10) Number of Ada variables generated.
11) Memory size of AG generated Ada software.

**Measurements specific to TAE+:**

1) Number of display items, such as stripcharts, stretchers, buttons, icons, text labels, scrolling windows, etc.
2) Number of colors used in the displays.
3) Number of different fonts used.
4) Number of display screens.
5) Number of display panels or windows.
6) Amount of information contained on a single screen - track for quality, there may be some user interface standards which state how much information should be presented to an operator on a single screen.
7) Number of data conversions required in order to obtain usable data on the display.
8) Number of variables, constants, TAE+ libraries required.
9) Total generated C LOC.
10) Number of TAE+ LOC which had to be modified to create a user interface screen.
11) Number of additional LOC not generated by TAE+ required to manipulate the GUI.
12) Total C commands.
13) Memory size required for libraries, executable code, graphical files, TAE+ software, X windows software, C software, and Open Look software, constants, and variables.

**Measurements specific to the developed Data Transfer Process:**

1) Developed C LOC.
2) Number of Data Transfer application variables.
3) Total C Commands.
4) Memory size of executable code.

**Computer System Measurements:**

1) Memory size of required RAM.
2) Size of disk storage per machine.
3) Size of swap space, if applicable.
4) Speed of processor.

**Tracking Requirements:**

Sequentially number (001, 002, ...) all requirements based on "shall" statements - some "shall" statements will contain more

than one requirement and should be broken down into multiple requirements.

When a requirement is modified it is assigned the next available sequential number. Old requirement numbers which have been modified should be flagged to track changes.

**Measurements specific to the Spiral Model:**

Track risks - risks are assigned to and tracked for each cycle, risks could be tracked based on elimination or a percentage of reduction. Measure percentage of risk reduction.

Demonstration Criteria - track passed criteria and failed criteria, track criteria carried over to delta-demos.

**Interfaces:**

Hardware to Hardware:

1) AG workstation to AG controller, via ethernet.
2) AG controller to AG I/O rack, via ethernet.
3) AG I/O rack to real hardware, via power box.
4) AG controller to SPARC workstation, via RS-232 cable.
5) Manual control box to real hardware.

Hardware to Software:

1) RS-232 port to Data Transfer application residing on SPARC workstation.
2) RS-232 port to AG controller software.
3) I/O rack to AG controller I/O processor.

Software to Software:

1) Data Transfer application to TAE+ code.
2) Data Transfer application to AG controller software.
3) X Windows to TAE+.
4) Open Look window manager to TAE+.
5) AG controller software subsystems (found in AG generated code).
6) AG controller software to Interactive Animation software residing on AG workstation.

**Project Changes:**

System Changes:

1) Heating of water - At first, water was pumped from Tank 1, through heater, and back into Tank 1. Currently, water is pumped from Tank 1, through

heater, and into Tank 2. This change occurred so that a process was in between two tanks and to better mimic Space Station Freedom.

2) Deletion of VAX GPX and inclusion of Sun 3/260. The VAX GPX was deleted from the data transfer loop because the AG did not communicate over DECnet but over ethernet, TCP/IP. The VAX GPX required a DECnet interface. The SUN workstation communicates over TCP/IP. The SPARC workstation also communicates over TCP/IP, but the Sun workstation was put into the loop to mimic Space Station Freedom architecture.
3) Inclusion of RS-232 serial communications cable for communication between the Sun 3/260 and AG.
4) Move Data Transfer application from Sun 3/260 to SPARC workstation. This change occurred because the Sun's hard drive crashed.
5) System became mode-oriented instead of setpoint-oriented. Modes controlled the system and the setpoints were utilized according to the current system mode.

Control changes:

1) The light control function was changed to a filter control function to better mimic a water recovery system.
2) Modifications were made to the water level equation which converted the water level sensor voltage to inches. The original equation's results did not match the actual water level of a tank.
3) Data transfer rate was increased from five seconds to three seconds.
4) Mode conditions changed as follows:

   Initialize mode - initial water levels changed to suit caution conditions instead of nominal system conditions.

   Standby mode - only used after the initialize mode. Before, system transitioned to standby if there was a system problem.

   Operate mode - changed from satisfying a level condition to satisfying a temperature and level condition before transitioning out of operate mode.

   Monitor mode - system transitions to monitor mode and operator has to select the initialize mode. Before, the transition to monitor mode only occurred when the water was "clean".

   Shutdown mode - only used to shut down the

system. Before, shutdown mode was used to shut down the system if there was a problem or the operation was over.

User Interface changes:

1) Variables and constants were added or deleted as necessary.
2) Routines to change indicator colors, change stripcharts, and turn items "on/off" were improved as the developer gained a better understanding of the TAE+ code.
3) Routines were added as the display functionality increased.
4) Changes were made based on customer feedback.

## 6.0 PROJECT DOCUMENTATION

The documentation developed for this project was impacted by several non-traditional aspects of this project. First, changes to the system and requirements were encouraged and expected during development of the prototype, therefore requirements changed and evolved during each cycle. Secondly, CASE tools and code generators were used to develop the system design and generate large portions of the software, eliminating the need to review design or test software. Thirdly, development of the prototype relied heavily on a series of demonstrations to evaluate the progress of the project. Based on these unique project aspects, the documentation developed for the Spiral Model Pilot project was tailored to meet project needs.

### 6.1 SMADR Documentation

The SMADR calls for the following documentation to be addressed:

| | |
|---|---|
| DM01 | Software Management Plan |
| DM06 | Software Requirements Specification |
| DM07 | Software Fault Tolerance and FMEA |
| DM09 | Detailed Software Design Specification (As Built) |
| DM16 | Software Schedule Document |
| DM20 | Software Verification Test Specification |
| DM23 | Software Verification Test Reports |
| DM28 | Configuration Data File Document |
| DM29 | Software Operator Manual |

The following sections discuss the development of project documentation and why some documents were not developed.

#### 6.1.1 Software Management Plan

The Software Management Plan (SMP), WP-2210/0025/0001/00, was developed during the planning phase of the project in concurrence with the risk definition, risk prioritization, and scheduling of project activities. A risk management approach was included in the SMP discussing the approach for determining and defining risks and specific development activities.

#### 6.1.2 Software Requirements and Design Specification

The Software Requirements Specification, WP-2210/0024/0001/01, was developed at the start of the project. A review of requirements occurred at the end of each cycle and updates made to the Requirements Specification as necessary. As the project cycled and evolved, design information was included in the specification creating a Software Requirements and Design Specification. Because this was a pilot project with a condensed project development time frame a separate Design Specification was not developed. The use of CASE tools and code generators also eliminated the need for a

Design Specification.

### 6.1.3 Software Fault Tolerance and FMEA

A Software Fault Tolerance and FMEA was not required due to development of a prototype. There were no programmatic risks of system failures or requirements for either fault tolerance or failure mode analysis for the Spiral Model Pilot project.

### 6.1.4 Detailed Software Design Specification

The Detailed Software Design Specification (As Built) was maintained electronically as TAE+ display screens and AG control block diagrams. Hardcopies of the TAE+ display screens, AG control block diagrams, C code generated by TAE+, Ada code generated from the AG control block diagrams, and the project developed Data Transfer application were delivered as an As Built appendix in the Software Requirements and Design Specification. The utilization of CASE tools and code generators alleviated the need for a separate Detailed Software Design Specification.

### 6.1.5 Software Schedule Document

A Software Schedule Document was not developed. The schedule for this project was included in the Life Cycle section of the SMP and a more detailed schedule is included in this document (see Table 4.1-1).

### 6.1.6 Software Verification Test Specification and Reports

The Software Verification Test Specification and Software Verification Test Reports were not developed, but were maintained as demonstrations' Pass/Fail Criteria. The Pass/Fail Criteria contained a test procedure to be followed at a demonstration and test results, including failed criteria or software discrepancies. Cycles three and four tested specific software requirements found in the Software Requirements and Design Specification. The requirement text and requirement numbers (document section numbers) were included in the Pass/Fail Criteria for demonstrations three and four.

### 6.1.7 Configuration Data File Document and Software Operator Manual

A Configuration Data File Document and a Software Operator Manual was not developed due to time and budget constraints. As mentioned earlier, this was a pilot project with a shortened project development time frame. Back-up tapes of all project files were delivered to the contracting officer and program manager, who also served as configuration managers. The customer was very involved in the development and demonstration of the prototype and, therefore, familiar with the operation of the prototype system.

## 6.2 Documentation Development

The purpose in the documentation developed for the Spiral Model Pilot project was to define the development process and display the completeness of project requirements and design. For any project, the documentation should be complete, fully define the system requirements and design, and define the development environment. Measuring the completeness of project documentation is a difficult task on all projects.

An iterative development of documentation, as permitted by the Spiral Model, allows the documentation to evolve as the system is developed. This helps to ensure that requirements and design are consistent with the system being developed.

The first draft of the Requirements Specification for the Spiral Model Pilot project consisted of information known at that time, not a lot of "To Be Determined" (TBDs). Requirements that were known were documented, developed, and tested. The first prototype demonstrated if requirements were practical and desirable by the customer. The review process allowed requirements to be changed where necessary and derived where missing.

It is important to understand that requirements were baselined at the start of a cycle. The development of a prototype based on a cycle's baselined requirements and design assisted in determining if the documentation was complete and adequate for that cycle. As the prototype evolved so did the documentation. Development of the prototype stopped when a product existed which satisfied the customer's expectations and needs. For the Spiral Model Pilot project, customer acceptability occurred in cycle five. Documentation was completed based on the prototype accepted by the customer.

## 7.0 PROJECT CONCLUSIONS

The following conclusions were drawn from an evaluation of this project.

1) The Spiral Model approach established an environment to not only allow but solicit modification of system requirements.

2) The Spiral Model allowed for the easy identification and implementation of changing system requirements to accommodate both resource changes and better understood requirements.

3) The project software documentation was not followed as diligently as under the standard waterfall approach. This has been attributed primarily to the focus on the prototype testbed. Utilization of CASE tools did not completely meet the needs for the documentation requirements.

4) The identification and use of the Pass/Fail Criteria allowed the entire team to focus on the desired characteristics of the final product.

5) The Spiral Model approach allowed for the demonstration of a maturing final product to all levels of management in a format that was easier to understand as compared to a detailed documentation review.

6) The Spiral Model approach encourages if not forces the team members to work closely together on a daily basis. The emphasis was on the exchange of ideas and information that was not handicapped by having to prepare documents that would not be used by anyone.

7) Utilization of CASE tools and code generators changed the need for certain "traditional" metrics. A review of recommended metrics needs to be re-addressed to accommodate technologies such as object-oriented and code generators.

8) Utilization of the Spiral Model does not give a good perspective of when the project is concluded. Continual modifications are always desirable or mandatory. Project management must make a value judgment of when the project is concluded, i.e. what is the last cycle.

9) This project did not provide any insight into the maintenance phase of projects and the applicability of the Spiral Model to that phase.

10) The subjective setting of risk values could lead to the setting of incorrect priorities to project risks.

11) The Spiral Model approach allows the development team to focus on smaller segments of code and to clearly define program objects, functions, interfaces, and operations.

12) From the development programmer's perspective, the Spiral Model demonstrations provided for very positive feedback of project status and therefore helped maintain high morale on the project.

# APPENDIX A

# DEMONSTRATION REPORTS

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Demo 1**
**May 31, 1991**

## Purpose

Demo 1 will display the successful networking of the AG controller to the Sun 3/260. The AG controller will also interface with the water recovery system hardware to read sensor values. Data will be transferred between the AG controller and Sun 3/260 and displayed at the Sun 3/260. The data will be displayed to determine that data is being transferred to the Sun 3/260. The Data Transfer application, to reside on the Sun 3/260, will eventually receive this data, however, the application will not be developed for this demo.

A vendor-provided demonstration of TAE+ capabilities will be executed to demonstrate the successful installation of TAE+ on the SPARC workstation, TAE+'s compatibility with other applications, such as Open Windows, on the SPARC workstation, and the text and graphical features of TAE+.

An initial Graphical User Interface (GUI) will be developed and demonstrated. A test data file will also be developed and used by the initial GUI to display test hardware sensor values. The test data file will provide the developer with an opportunity to experiment with modifying the TAE+ C code to receive parameter values for display.

## Risks To Be Addressed and Planned Percentage of Completion

1) Networking/Integration - 100% complete
   Interface AG controller to Sun 3/260 - (100%).

2) Graphical User Interface - 48% complete
   Completion of TAE+ installation (5%).
   Software applications compatibility on SPARC workstation (5%).
   Successful modification of TAE+ C code to accept hardware data (30%).
   GUI acceptability (8%).

3) Hardware Interface - 40% complete
   Interface AG controller to water recovery system hardware (40%).

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated during Demo 1, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria will be represented by an "x" in the space provided.

**Check if test passes**

Function 1 - Networking/Integration.

___ 1) Display AG controller connectivity to serial communications cable.

Comments:

___ 2) Display Sun 3/260 connectivity to serial communications cable.

Comments:

___ 3) Display AG controller data at Sun 3/260 in a readable (ascii) format. Data should also be displayed at the AG workstation (interactive animation) or stored in a data file at the AG workstation and compared to data displayed at Sun 3/260. This is to demonstrate that data is not changed when it is transferred to the Sun 3/260.

Comments:

Function 2 - TAE+ Installation Complete.

___ 1) Provide a TAE+ vendor-supplied demo demonstrating TAE+ text and graphical features.

Comments:

Function 3 - Software Applications Compatibility on SPARC WS.
Function 4 - TAE+ C Code Modifications.
Function 5 - Graphical User Interface Acceptability.

___ 1) Display TAE+ use of multiple windows (Open Windows).

Comments:

_____

_____

2) Display a test TAE+ user interface displaying all AG transmitted values and current controller software state:

___ - controller software state
___ - water level of Tank 1
___ - water level of Tank 2
___ - water temperature of Tank 2
___ - state of Pump 1
___ - state of Pump 2
___ - state of Pump 3
___ - state of Pump 4
___ - state of Pump 5
___ - state of heater
___ - state of light
___ - alarm state of Tank 1
___ - alarm state of Tank 2
___ - alarm state of Reservoir.

Comments:

_____

_____

___ 3) Display local test data file.

Comments:

_____

_____

___ 4) Display modified TAE+ C code with inclusion of local test data file.

Comments:

_____

_____

___ 5) Display test data read by test display for at least one icon. Icon outputs should be consistent with data contained in local test data file.

Comments:

_____

_____

___ Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

_____

_____

**Demo Results**

Demo 1 was very unsuccessful. The Pass/Fail Criteria was presented to the developers late in the cycle and there was misunderstanding between the developers and management as to the purpose of the Pass/Fail Criteria. Demo 1 was failed and a Delta-Demo scheduled for one week later to repeat the demo's Pass/Fail Criteria. The scheduling of a Delta-Demo did not slide the scheduling of later demos.

It was determined that a Pass/Fail Criteria needed to be provided to developers two weeks prior to a demonstration.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Delta-Demo 1**
**June 7, 1991**

**Purpose**

Same as Demo 1.

**Risks To Be Addressed and Planned Percentage of Completion**

Same as Demo 1.

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated during Delta-Demo 1, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria will be represented by an "x" in the space provided.

**Check if test passes**

Function 1 - Networking/Integration.

_X_ 1) Display AG controller connectivity to serial communications cable.

Comments:

_____

_____

_X_ 2) Display Sun 3/260 connectivity to serial communications cable.

Comments:

_____

_____

_x_ 3) Display AG controller data at Sun 3/260 in a readable (ascii) format. Data should also be displayed at the AG workstation (interactive animation) or stored in a data file at the AG workstation and compared to data displayed at Sun 3/260. This is to demonstrate that data is not changed when it is transferred to the Sun 3/260.

Comments:

_____

_____

Function 2 - TAE+ Installation Complete.

_x_ 1) Provide a TAE+ vendor-supplied demo demonstrating TAE+ text and graphical features.

Comments:

_____

_____

Function 3 - Software Applications Compatibility on SPARC WS.
Function 4 - TAE+ C Code Modifications.
Function 5 - Graphical User Interface Acceptability.

_x_ 1) Display TAE+ use of multiple windows (Open Windows).

Comments:

_____

_____

2) Display a test TAE+ user interface displaying all AG transmitted values and current controller software state:

_x_ - controller software state
_x_ - water level of Tank 1
_x_ - water level of Tank 2
_x_ - water temperature of Tank 2
_x_ - state of Pump 1
_x_ - state of Pump 2
_x_ - state of Pump 3
_x_ - state of Pump 4
_x_ - state of Pump 5
_x_ - state of heater
_x_ - state of light
_x_ - alarm state of Tank 1
_x_ - alarm state of Tank 2
_x_ - alarm state of Reservoir.

Comments:
Data displayed in GUI was pre-assigned to icons, data was not passed into GUI as parameters.

x 3) Display local test data file.

Comments:

___ 4) Display modified TAE+ C code with inclusion of local test data file.

Comments:
Capability was not available for testing due to a time constraint.

x 5) Display test data read by test display for at least one icon. Icon outputs should be consistent with data contained in local test data file.

Comments:

x Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

Customer would like to see a panel that represents the system showing tanks, pumps, heater, etc. Font for temperature value needs to be larger. Alarm messages should change colors.

**Demo Results**

All criteria was passed except for the modification of TAE+ C code and the inclusion of test data in the TAE+ graphical user interface (GUI). This criteria failed due to a time constraint, not an unsuccessful implementation of the criteria. It was agreed that the display of this function could be delayed until Demo 2 and at Demo 2 real hardware data would be included in the GUI instead of test data.

At the end of cycle one, the networking/integration risk was eliminated by 100% and the hardware interface risk reduced by 40% as planned. The GUI acceptability risk was scheduled to be reduced by 48%. This risk was only reduced by 18% because test data had not been included in the GUI for display and the TAE+ C code had not been modified.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Demo 2**
**June 28, 1991**

**Purpose**

Demo 2 will display an interface between the AG controller and the water recovery hardware. The final control algorithms, which will include logic conditions and state controls, will not be developed. Hardware states, however, will be controlled (activated/deactivated) and monitored via AG generated software. Hardware states will be displayed on the AG interactive animation. The hardware will also be manually controlled via the manual control box and hardware sensor values monitored on the AG interactive animation.

The Data Transfer application will be developed and implemented. The Data Transfer application will be developed in C, reside on the Sun 3/260, and transfer real hardware data from the AG controller to the SPARC workstation, via the Sun 3/260.

The graphical user interface (GUI) will be displayed for customer feedback. The GUI will include and display real hardware data output. The output may be limited to one analog sensor data, one discrete sensor data, and the system's model of operation. This test will display the inclusion of each type of data which will be transmitted by the AG controller.

**Risks To Be Addressed and Planned Percentage of Completion**

1) Graphical User Interface - 76%
   Successful modification of TAE+ C code to accept hardware data (30%).
   Display of real hardware data at the GUI (20%).
   GUI acceptability (8%).

2) Hardware Interface - 100%
   Develop Data Transfer application to transfer data from AG controller to SPARC workstation, via Sun 3/260 (50%).
   Display hardware data transferred to Sun 3/260 and SPARC workstation (10%).

3) Development of Control Algorithms - 40%
   Develop algorithms to activate actuators and read sensors (40%).
   Exercise hardware via manual control box and read sensor values.

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated for Demo 2, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria will be represented by an "x" in the space provided.

**Check if test passes**

Function 1 - AG Controller and Hardware Interface.

a) Exercise Water Recovery System hardware via AG controller software.

_x_ - Display AG interactive animation on AG workstation.
_x_ - Generate Ada software at AG workstation.
_x_ - Compile and link Ada software at AG workstation.
_x_ - Download Ada software to AG controller from AG ws.
_x_ - Display I/O connections between HW and AG I/O rack.

Comments:

b) Acquire Water Recovery System sensor data and display on AG interactive animation. Data to be collected and displayed:

___ - water level of Tank 1
___ - water level of Tank 2
___ - water temperature of Tank 2
___ - state of Pump 1 In
___ - state of Pump 1 Out
___ - state of Pump 2 In
___ - state of Pump 2 Out
___ - state of Pump Heater
___ - state of heater
___ - state of filter
___ - alarm state of Tank 1
___ - alarm state of Tank 2
___ - alarm state of Reservoir.

Comments:

x c) Display Water Recovery System mode of operation on AG interactive animation.

Comments:
Mode of operation was simulated. This is acceptable, mode control implementation is scheduled for Demo 3.

x d) Exercise Water Recovery System via the manual control box.

Comments:
During test, a problem with the reservoir's high alarm LED was discovered. NRC will investigate problem.

Function 2 - Data Transfer from Hardware to SPARC workstation.

___ a) Display Water Recovery System sensor data at Sun 3/260 in a real-time, readable format or in a stored, readable data file.

Comments:

b) Display Water Recovery System data in the graphical user interface residing on SPARC workstation:

___ - one analog sensor data
___ - one discrete sensor data
___ - system's mode of operation.

Comments:

x c) Display C application (Data Transfer Application) residing on Sun 3/260 and application's access of AG controller or hardware data.

Comments:

___ d) Display modified TAE+ C code, residing on SPARC workstation, and code's access of data output by the Data Transfer Application residing on Sun 3/260.

Comments:
Criteria was not available for testing due to a time constraint.

___ Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

Information that is displayed in more than one way is not necessary, the screen appears to be too crowded. Only display information once. Strip charts are too large, causing the operator's vision to drift toward them. Text needs to be larger so it can be read more easily. Tank and pipe icons tend to fade into the background, they need to stand out more. Colors used in display are very good.

**Demo Results**

Demo 2 successfully displayed an interface between the AG controller and water recovery hardware and between the manual control box and hardware. The AG controller software or control algorithms developed for demo 2 allowed the operator to change the states of hardware actuators but did not provide the capability to read or monitor incoming hardware sensor data. The sensor data read at the AG workstation was simulated, not hardware driven.

The Data Transfer application was developed and accessed AG controller data. However, the data transferred between the AG controller and SPARC workstation, via the Sun 3/260, was simulated data, not real hardware data. The GUI did not display any sensor data. Modification of TAE+ code to access hardware data was not implemented due to time constraints.

A delta-demo will be held in two weeks (7/12/91) to demonstrate the AG controller reading incoming hardware signals, transfer of real hardware data between the AG controller and SPARC workstation, GUI display of simulated or real hardware sensor data, and a modification of TAE+ C code to access hardware data. The delta-demo will not impact the scheduling of future demos.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Delta-Demo 2**
**July 12, 1991**

## Purpose

Delta-demo 2 will demonstrate Demo 2's failed criteria. The water recovery hardware sensor data will be displayed on the AG interactive animation, the Sun 3/260 workstation, and the SPARC workstation, demonstrating the complete implementation of the Data Transfer application. The TAE+ C code will be modified to access real hardware data. At a minimum, the graphical user interface (GUI) will display one analog sensor data value, one discrete sensor data value, and the system's mode of operation. If development time permits, real hardware data will be displayed at the GUI else the data displayed will be simulated.

## Risks To Be Addressed and Planned Percentage of Completion

1) Graphical User Interface - 76%
   Successful modification of TAE+ C code to accept hardware data.
   Display sensor data at GUI.

2) Hardware Interface - 100%
   Complete Data Transfer application to transfer real hardware data from AG controller to SPARC workstation, via Sun 3/260.
   Display hardware data transferred to Sun 3/260 and SPARC workstation.

3) Development of Control Algorithms - 40%
   Complete algorithms to read hardware sensor values.

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated for Delta-Demo 2, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria will be represented by an "x" in the space provided.

**Check if test passes**

Function 1 - AG Controller and Hardware Interface.

a) Acquire Water Recovery System sensor data and display on AG interactive animation. Data to be collected and displayed:

| | |
|---|---|
| x | - water level of Tank 1 |
| x | - water level of Tank 2 |
| x | - water temperature of Tank 2 |
| x | - state of Pump 1 In |
| x | - state of Pump 1 Out |
| x | - state of Pump 2 In |
| x | - state of Pump 2 Out |
| x | - state of Pump Heater |
| x | - state of heater |
| x | - state of filter |
| x | - alarm state of Tank 1 |
| x | - alarm state of Tank 2 |
| x | - alarm state of Reservoir. |

Comments:
Sensor data displayed at AG interactive animation was simulated data, not hardware-driven data.

Function 2 - Data Transfer from Hardware to SPARC workstation.

x a) Display Water Recovery System sensor data at Sun 3/260 in a real-time, readable format or in a stored, readable data file.

Comments:
Data displayed at Sun 3/260 was simulated data, not hardware-driven data.

b) Display Water Recovery System data in the graphical user interface residing on SPARC workstation:

x - one analog sensor data
x - one discrete sensor data
x - system's mode of operation.

Comments:
Data displayed was built into GUI, data was not accessed by GUI or passed in as a parameter.

___ c) Display modified TAE+ C code, residing on SPARC workstation, and code's access of data output by the Data Transfer Application residing on Sun 3/260.

Comments:

x Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

**Demo Results**

All delta-demo 2 criteria passed except for modification of the TAE+ C code to access hardware data. It was determined that the development schedule was too ambitious and that the modification of TAE+ C code could be demonstrated at Demo 3. The customer agreed that the GUI displayed at the delta-demo was acceptable for Demo 3 and would not require any changes. This will allow the GUI developer to devote their time to modifying the TAE+ C code to access water recovery hardware data.

The GUI risk was planned to be 76% complete, however, the risk was only reduced by 54% due to the modification of TAE+ C code not being implemented. The GUI was accepted for cycle 2 and 3 making up 16% of the 54% reduction.

The Hardware Interface risk was scheduled for 100% completion, however, displayed sensor data at the AG interactive animation, Sun 3/260, and SPARC workstation was simulated data, not hardware driven, thus this risk is only 90% reduced. Control algorithms were to be developed to activate actuators and read sensor values. Because sensor values were simulated, this risk was reduced by 30% instead of 40%.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Demo 3**
**August 2, 1991**

## Purpose

Demo 3 will present the Water Recovery Control System, demonstrating all requirements as defined in the ECLSS Process Control Prototype Requirements and Design Specification. The graphical user interface (GUI) will display all water recovery hardware activity including hardware states, fluctuations in temperature and water levels, caution and warning alarm conditions, and the system's mode of operation.

## Risks To Be Addressed and Planned Percentage of Completion

1) Development of Control Algorithms - 100%
   Final and complete control algorithm development.

2) Graphical User Interface - 84%
   Successful modification of TAE+ C code to accept hardware data (30%).
   Display water recovery hardware activity at GUI.

## Pass/Fail Criteria

The following Pass/Fail Criteria defines the capability to be demonstrated for Demo 3, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. The highlighted text represents requirements from the Requirements and Design Specification. Document section numbers are also included. The operator action describes how the operator will test a criteria. A passed criteria will be represented by an "x" in the space provided.

**Check if test passes**

### 3.1.3.6.1 Monitor Mode Requirements

User shall specify the following set point values at the AG workstation's interactive animation:

Tank 1 Water Level Set Point
Tank 2 Water Level Set Point
Tank 2 Temperature Set Point

3.1.3.1(1) Read and store operator set points for Tank 1 and Tank 2 water levels.

3.1.3.1(2) Read and store operator set point for Tank 2 water temperature.

3.1.3.5(8) Display Tank 1 and Tank 2 water level set points.

3.1.3.5(9) Display Tank 2 temperature set point.

User shall control the following actuators at the AG workstation's interactive animation:

Tank 1 Input Pump
Tank 1 Output Pump
Tank 2 Input Pump
Tank 2 Output Pump
Heater Pump
Heater
Filter

3.1.3.1(4) Read and store operator selected pumps to be deactivated.

3.1.3.1(3) Read and store operator selected state of filter.

3.1.3.4(1) Turn filter on if operator selected state of filter is on.

3.1.3.4(2) Turn filter off if operator selected state of filter is off.

The Display Function shall dynamically display all sensor values.

3.1.3.5(1) Display Tank 1 and Tank 2 water levels in inches.

3.1.3.5(2) Display state of Tank 1, Tank 2, and Reservoir water level limit alarms.

3.1.3.5(3) Display speed of all pumps.

3.1.3.5(4) Display if a pump is activated or deactivated.

3.1.3.5(5) Display Tank 2 water temperature in degrees fahrenheit.

3.1.3.5(6) Display state of heater.

3.1.3.5(7) Display state of filter.

Message is displayed requesting operator to deactivate actuators

prior to mode transitioning and resolve any alarm conditions.

Operator is able to transition to Initialize Mode if all actuators are deactivated and no alarm conditions exist.

Operator is not able to transition to Standby Mode and Operate Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

Operator Action:

Operator inputs Tank 1 and Tank 2 water level set points at AG workstation's interactive animation.

Operator inputs Tank 2 temperature set point at AG workstation's interactive animation (set point value is less than current Tank 2 temperature).

Criteria:

x Operator is able to input Tank 1 and Tank 2 water level set points at AG workstation's interactive animation.

x Tank 1 and Tank 2 water level set point values are displayed at AG workstation's interactive animation.

x Operator is able to enter Tank 2 temperature set point at AG workstation's interactive animation.

x Tank 2 temperature set point is displayed on interactive animation.

Operator Action:

Operator selects speed (0.00 for off) at AG workstation's interactive animation for pumps.

Operator selects state of Heater.

Operator selects state of Filter.

Criteria:

x Operator is able to select speed of all five pumps.

x Operator is able to select state of Heater.

x Operator is able to select state of Filter.

x Tank 1 and Tank 2 water levels are displayed in inches.

x Tank 1, Tank 2, and Reservoir water level limit alarms are displayed.

Speed and state (0.00 for off) of the following pumps are displayed:

x Tank1 Input Pump
x Tank1 Output Pump
x Tank2 Input Pump
x Tank2 Output Pump
x Heater Pump

x Tank 2 water temperature is displayed in degrees fahrenheit.

x State of heater is displayed.

x State of filter is displayed.

x Message is displayed requesting operator to deactivate actuators prior to mode transitioning and resolve any alarm conditions.

Operator Action:

Operator attempts to transition to Initialize Mode when an actuator is on or an alarm condition exists.

Criteria:

x Operator is not able to transition to Initialize Mode.

Operator Action:

Operator attempts to transition to Initialize Mode when all actuators are off and there are no alarm conditions.

Criteria:

x Operator is able to transition to Initialize Mode.

Operator Action:

Operator attempts to transition to Standby Mode.

Operator attempts to transition to Operate Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

x Operator is not able to transition to Standby Mode.

_x_ Operator is not able to transition to Operate Mode.

_x_ Operator is not able to transition to Shutdown Mode.

### 3.1.3.6.2 Initialize Mode Requirements

AG controller software achieves Tank 1 water level set point value defined in Monitor Mode.

AG controller software achieves Tank 2 water level set point value defined in Monitor Mode.

3.1.3.2(1) Continuously read Tank 1 and Tank 2 height measurements in voltage and convert measurements to inches...

3.1.3.2(2) Continuously store Tank 1 and Tank 2 water level measurements in inches.

3.1.3.2(3) Continuously compare Tank 1 and Tank 2 actual water levels to Tank 1 and Tank 2 water level set points.

3.1.3.2(4) Control Tank 1 and Tank 2 Input and Output pumps to achieve the water level within a deadband of one inch.

AG controller software transitions to Standby Mode when Tank 1 and Tank 2 water level set point values are achieved.

Operator is able to transition to Monitor Mode.

Operator is not able to transition to Operate Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

3.1.3.6 For any mode if an alarm condition exists, the AG controller software deactivates all actuators and transitions to Monitor Mode.

3.1.3.2(5) Continuously read high level and low level inputs for Tank 1, Tank 2, and Reservoir to determine if an alarm condition exists.

3.1.3.2(6) Stop all pumps if an alarm condition exists.

Criteria:

_x_ Display of Tank 1 and Tank 2 water levels is consistent with real tank water levels.

_x_ Display of Tank 1 and Tank 2 water levels is in inches.

_x_ AG controller software achieves operator set point value for Tank 1.

_x_ AG controller software achieves operator set point value for Tank 2.

_x_ AG controller software transitions to Standby Mode when Tank 1 and Tank 2 water level set point values are achieved.

Operator Action:

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Operate Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

___ Operator is able to transition to Monitor Mode.

___ Operator is not able to transition to Operate Mode.

___ Operator is able to transition to Shutdown Mode.

Operator Action:

Create an alarm condition via manual control box.

Criteria:

_x_ AG controller software deactivates all actuators and transitions to Monitor Mode.

**3.1.3.6.3 Standby Mode**

AG controller software deactivates all Water Recovery pumps.

When actual tank water levels do not stay at within a deadband of one inch (+/- 1/2 inch) for water level set point values, the AG controller software transitions to Initialize Mode.

Operator can select if AG controller software automatically transitions to Operate Mode of if manual transition to Operate Mode is required.

AG controller software automatically transitions to operate mode if automatic transmission is set to on.

Operator is able to transition to Operate Mode.

Operator is able to transition to Monitor Mode.

Operator is not able to transition to Initialize Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

3.1.3.6 For any mode if an alarm condition exists, the AG controller software deactivates all actuators and transitions to Monitor Mode.

3.1.3.2(5) Continuously read high level and low level inputs for Tank 1, Tank 2, and Reservoir to determine if an alarm condition exists.

3.1.3.2(6) Stop all pumps if an alarm condition exists.

Criteria:

_x_ Upon transitioning to Standby Mode, all pumps are deactivated.

Operator Action:

Manually, via manual control box, change Tank 1 and/or Tank 2 actual water levels so that water levels are outside the one inch deadband for set point values.

Criteria:

_x_ AG controller software transitions to Initialize Mode.

Operator Action:

Operator sets auto transition to Operate Mode to on.

Criteria:

_x_ AG controller software automatically transitions to Operate Mode.

Operator Action:

Operator set auto transition to Operate Mode to off.

Criteria:

_x_ AG controller software waits for operator' selection of Operate Mode before transitioning to Operate Mode.

Operator Action:

Operator attempts to transition to Operate Mode.

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Initialize Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

___ Operator is able to transition to Operate Mode.

___ Operator is able to transition to Monitor Mode.

___ Operator is not able to transition to Initialize Mode.

___ Operator is able to transition to Shutdown Mode.

Operator Action:

Create an alarm condition via manual control box.

Criteria:

___ AG controller software deactivates all actuators and transitions to Monitor Mode.

### 3.1.3.6.4 Operate Mode Requirements

AG controller software activates heater pump.

AG controller software attempts to achieve temperature set point value defined in Monitor Mode.

| | |
|---|---|
| 3.1.3.3(1) | Continuously read Tank 2 water temperature in voltage and convert voltage to fahrenheit... |
| 3.1.3.3(2) | Continuously store Tank 2 water temperature in fahrenheit. |
| 3.1.3.3(3) | Compare Tank 2 water temperature to water temperature set point. |
| 3.1.3.3(4) | If water temperature is less than temperature set point, turn heater and heater pump on and pump water through the heater to maintain the temperature at or above the temperature set point. |

AG controller software shall activate the filter system.

AG controller software transitions to Monitor Mode when (Tank 1 Empty is met or Tank 2 Full is met) and (Temperature set point is met (actual temperature >= temperature set point value)).

AG controller software transitions to Initialize Mode when (Tank 1 Empty is met or Tank 2 Full is met) and (Temperature set point is not met (actual temperature < temperature set point value)).

Operator is not able to transition to Monitor Mode, Initialize Mode, and Standby Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

3.1.3.6 For any mode if an alarm condition exists, the AG controller software deactivates all actuators and transitions to Monitor Mode.

3.1.3.2(5) Continuously read high level and low level inputs for Tank 1, Tank 2, and Reservoir to determine if an alarm condition exists.

3.1.3.2(6) Stop all pumps if an alarm condition exists.

Criteria:

___ Tank 2's water temperature is displayed in degrees fahrenheit on the AG workstation's interactive animation.

___ Changes in Tank 2's water temperature are displayed on interactive animation.

_x_ AG controller software turns heater and heater pump on.

___ AG controller software achieves and maintains a Tank 2 temperature at or above the temperature set point value.

_x_ AG controller software activates the filter system.

Operator Action:

Operator achieves a Tank 1 Empty condition (Tank 1 water level = 4 inches) or a Tank 2 Full condition (Tank 2 water level = 14 inches) via manual controls when Temperature set point is met.

Criteria:

_x_ AG controller software transitions to Monitor Mode.

Operator Action:

Operator achieves a Tank 2 Empty condition (Tank 1 water level = 4 inches) or a Tank 2 Full condition (Tank 2 water level = 14

inches) via manual controls when Temperature set point is not met.

Criteria:

_x_ AG controller transitions to Initialize Mode.

Operator Action:

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Initialize Mode.

Operator attempts to transition to Standby Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

___ Operator is not able to transition to Monitor Mode.

_x_ Operator is not able to transition to Initialize Mode.

___ Operator is not able to transition to Standby Mode.

___ Operator is able to transition to Shutdown Mode.

Operator Action:

Create an alarm condition via manual control box.

Criteria:

_x_ AG controller software deactivates all actuators and transitions to Monitor Mode.

### 3.1.3.6.5 Shutdown Mode

AG controller software deactivates all actuators.

Data stream to Data Transfer Process residing on Sun workstation is stopped (soft stop).

Operator is not able to transition to any other mode.

Operator Action:

Operator attempts to transition to another mode.

Criteria:

___ AG controller software deactivates all actuators.

___ Data stream to Data Transfer Process is stopped.

___ Operator is not able to transition to another mode.

_x_ Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

Placement of filter makes it difficult to associate the filter with the Reservoir, Tank 1, or Tank 2. Would like to see water flowing through pipes when a pump is activated, if this is not possible arrows indicating the direction of water flow would be helpful. White lettering is easier to read than black lettering. Mode panel appears to have six modes instead of five modes. Overall the customer is very satisfied with the GUI.

**Demo Results**

Most criteria for demo 3 passed. Not all criteria demonstrating the correct mode transitions were tested due to a limited amount of time that demo participants could be available. It was agreed that mode transitions which were not tested could be demonstrated at demo 4.

Display of Tank 2's water temperature failed due to a faulty thermocouple. The selected thermocouple was inappropriate for liquid use. The criteria which failed was not due to incorrect control algorithm development, therefore the control algorithms were viewed as being correct and complete and the control algorithm development risk 100% eliminated. The thermocouple may not be replaced for this project. The final interface to the hardware also permitted hardware data to be transferred to the Sun 3/260 and SPARC workstation eliminating the hardware interface risk 100%.

The TAE+ C code was successfully modified to accept water recovery hardware data and display the data in the GUI. The actual percentage of risk reduction in this area had fallen behind due to a delay in modification of the TAE+ C code. The project had planned to reduce the GUI acceptability risk by 84% for cycle three and this reduction was achieved at demo 3. The customer was satisfied with the GUI display except for some changes which will be incorporated for demo 4.

The project was back on schedule at the end of cycle three. Earlier delta-demos did not affect the overall development schedule.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Demo 4**
**August 20, 1991**

**Purpose**

Demo 4 will present the final graphical user interface (GUI) for the Water Recovery Monitoring system, demonstrating all requirements as defined in the ECLSS Process Control Prototype Requirements and Design Specification. Demo 4 will also present the Water Recovery Control System criteria which were not displayed at Demo 3.

**Risks To Be Addressed and Planned Percentage of Completion**

1) Graphical User Interface - 92%
GUI acceptability (8%).

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated for Demo 4, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria will be represented by an "x" in the space provided.

The highlighted text represents requirements from the Requirement and Design Specification. Document section numbers are also included. The operator action describes how the operator will test a criteria.

**Check if test passes**

**Water Recovery Control System**

Due to time constraints, the following Water Recovery Control System criteria were not tested for Demo 3, held 8/2/91, and, therefore, did not pass. As part of Demo 4, the following will be tested.

**3.1.3.6.2 Initialize Mode Requirements**

Operator is able to transition to Monitor Mode.

Operator is not able to transition to Operate Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

Operator Action:

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Operate Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

_x_ Operator is able to transition to Monitor Mode.

_x_ Operator is not able to transition to Operate Mode.

_x_ Operator is able to transition to Shutdown Mode.

**3.1.3.6.3 Standby Mode**

Operator is able to transition to Operate Mode.

Operator is able to transition to Monitor Mode.

Operator is not able to transition to Initialize Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

3.1.3.6 For any mode if an alarm condition exists, the AG controller software deactivates all actuators and transitions to Monitor Mode.

3.1.3.2(5) Continuously read high level and low level inputs for Tank 1, Tank 2, and Reservoir to determine if an alarm condition exists.

3.1.3.2(6) Stop all pumps if an alarm condition exists.

Operator Action:

Operator attempts to transition to Operate Mode.

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Initialize Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

x Operator is able to transition to Operate Mode.

x Operator is able to transition to Monitor Mode.

x Operator is not able to transition to Initialize Mode.

x Operator is able to transition to Shutdown Mode.

Operator Action:

Create an alarm condition.

Criteria:

x AG controller software deactivates all actuators and transitions to Monitor Mode.

**3.1.3.6.4 Operate Mode Requirements**

Operator is able to transition to Monitor Mode.

Operator is not able to transition to Initialize Mode and Standby Mode.

3.1.3.6 Operator is able to transition to Shutdown Mode.

Operator Action:

Operator attempts to transition to Monitor Mode.

Operator attempts to transition to Standby Mode.

Operator attempts to transition to Shutdown Mode.

Criteria:

x Operator is able to transition to Monitor Mode.

x Operator is not able to transition to Standby Mode.

x Operator is able to transition to Shutdown Mode.

**3.1.3.6.5 Shutdown Mode**

AG controller software deactivates all actuators.

Data stream to Data Transfer Process residing on Sun workstation is stopped (soft stop).

Operator is not able to transition to any other mode.

Operator Action:

Operator attempts to transition to another mode.

Criteria:

x AG controller software deactivates all actuators.

x Data stream to Data Transfer Process is stopped.

x Operator is not able to transition to another mode.

**Water Recovery Monitoring System**

The following requirements are the criteria to be tested and demonstrated by display on the graphical user interface. No operator action is required.

___ All data displayed on the SPARC workstation's graphical user interface is consistent with water recovery system's hardware activity.

Comment:
Data displayed at the GUI is off by about 5 seconds. Developer will try to improve timing.

| | | |
|---|---|---|
| x | 3.2.3(1) | Read/Write the incoming real-time water recovery data. |
| x | 3.2.3(2) | Indicate the current mode of operation. |
| x | 3.2.4 | The mode button icon shall change color (green) when the mode becomes active. |
| x | 3.2.3(3) | Report caution/warning alarms for Tank 1, Tank 2, and Reservoir. |
| x | 3.2.4 | There shall be separate alarm indicators for high water level alarms and low water level alarms. |
| x | 3.2.4 | The caution and warning alarms shall be represented by button icons which change color to indicate caution (yellow) and warning (red). |

| | | |
|---|---|---|
| x | 3.2.4 | The alarm indicator shall remain the same color until the alarm condition is no longer existing. |
| | 3.2.4 | Caution alarms shall be given for Tank 1 and Tank 2 at the following threshold values: |
| x | | Tank 1 Low Level: 2.1 - 3.8 |
| x | | Tank 1 High Level: 14.0 - 15.7 |
| x | | Tank 2 Low Level: 2.1 - 3.8 |
| x | | Tank 2 High Level: 13.7 - 15.4 |
| ___ | 3.2.4 | Warning alarms shall be provided for Tank 1, Tank 2, and Reservoir. |
| x | 3.2.3(5) | Indicate current water levels for Tank 1 and Tank 2 in inches. |
| x | 3.2.4 | Tank 1 and Tank 2 icons shall dynamically indicate, at all times, the water levels. |
| x | 3.2.4 | The water level value of Tank 1 shall be available by clicking on the Tank 1 icon. |
| x | 3.2.4 | The water level value of Tank 2 shall be available by clicking on the Tank 2 icon. |
| x | 3.2.4 | The "water" in Tank 1 and Tank 2 shall change color to indicate alarm conditions (yellow for caution, and red for warning) or nominal conditions (blue). |
| x | 3.2.3(6) | Indicate status (activated or deactivated) for the following pumps: Tank1_In_Pump, Tank1_Out_Pump, Tank2_In_Pump, Tank2_Out_Pump, Heater_Pump. |
| x | 3.2.3(7) | Display the current speed of each pump. |
| x | 3.2.4 | The five pumps shall indicate whether they are in an active or inactive state by changing color (black arrow - inactive, green arrow - active). |
| x | 3.2.4 | The user shall be able to obtain a pump speed by clicking on the desired pump icon. |
| x | 3.2.3(8) | Convert Tank 2 temperature from degrees fahrenheit to degrees celsius. |
| x | 3.2.3(9) | Indicate current water temperature of Tank 2 in degrees celsius. |
| x | 3.2.4 | The thermocouple shall be represented by a thermometer with a range of 10 - 50 degrees celsius. |

3.2.4 The thermometer shall change color to indicate a change in temperature as follows:

x dodger blue - very cold
_ light blue - cold
x light pink - warm
x light red - medium warm
x red - hot

x 3.2.3(11) Indicate current status (on/off) of the filter.

x 3.2.4 The filter shall be located in the Reservoir and shall change color (black - inactive, green - active) and position when the sensor value is greater than 0.0.

x 3.2.3(12) Indicate current status (on/off) of the heater.

x 3.2.4 Heater shall change color (black - inactive, green - active) when the sensor value is greater than 0.0.

x 3.2.3(4) Provide the user with a means of monitoring temperature, tank levels, and flow rate changes over time.

x 3.2.4 There shall be a stripchart to represent Tank 1 level, Tank 2 level, temperature fluctuations, and flowrate over time. The stripcharts shall have the same threshold settings for color as their respective system representatives.

x Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

A label needs to be added for the filter. Pump speeds are currently represented as "-" and "+", these will be replaced with "min." and "max.". Arrows need to be put above pipes displaying the directional flow of water. The thermostat displays a different color for too many thresholds, the colors and thresholds will be reduced.

**Demo Results**

All Water Recovery Control System criteria from Demo 3 were passed. The customer was very satisfied with the GUI with the exception of a few items as noted in the comments section. There is approximately a five second delay between the data displayed at the GUI and the actual hardware activity. This timing issue will be resolved by Demo 5. Caution/warning alarms were not

activated/deactivated according to hardware activity. The GUI acceptability risk was reduced by only 90% due to failed criteria. Failed criteria will be presented at Demo 5 with GUI updates based on the customer's comments.

**SPIRAL MODEL PILOT PROJECT**
**ECLSS PROCESS CONTROL PROTOTYPE**

**Demo 5**
**September 9, 1991**

**Purpose**

Demo 5 will present the complete ECLSS Process Control Prototype. The Water Recovery Control System, Water Recovery Monitoring System, and Data Transfer Process will be tested at a functional system level. The graphical user interface (GUI) will be reviewed for final customer acceptance. GUI criteria failed at Demo 4 will be retested.

**Risks To Be Addressed and Planned Percentage of Completion**

1) Graphical User Interface - 100%
GUI acceptability (8%).

**Pass/Fail Criteria**

The following Pass/Fail Criteria defines the capability to be demonstrated for Demo 5, the approach for measuring the success of the developed capability, and a status of Pass/Fail defining the quality of the capability. A passed criteria is represented by an "x" in the space provided.

**Check if test passes**

**Water Recovery Monitoring System**

The following Water Recovery Monitoring System criteria did not pass for Demo 4 and will be tested as part of Demo 5.

_x_ All data displayed on the SPARC workstation's graphical user interface is consistent with water recovery system's hardware activity. (For Demo 4, timing was off by approximately 5 - 10 seconds.)

_x_ 3.2.4 Warning alarms shall be provided for Tank 1, Tank 2, and Reservoir. (For Demo 4, warning alarms were not working correctly.)

**ECLSS Process Control Prototype**

The following criteria are system level requirements for the ECLSS Process Control Prototype. All lower level requirements have been tested in previous demos.

_x_ The ECLSS Process Control Prototype shall control, via manual controls or controller software, the water recovery hardware system. (Water Recovery Control System)

_x_ The ECLSS Process Control Prototype shall monitor the hardware activity data, sensor data, and current system control state at a monitoring station. (Water Recovery Monitoring System)

_x_ The hardware activity data shall be transmitted over a network and displayed in real-time at the monitoring station.

_x_ Backup disks of all developed AG, Sun 3/260, and SPARC files provided to Contracting Officer and Project Manager.

Customer's comments on graphical user interface:

GUI is satisfactory.

**Demo Results**

All criteria passed and the ECLSS Process Control Prototype, including GUI, was accepted by the customer. The GUI acceptability risk was eliminated by 100%